为 VB设计一个消息控件

杨 强[20] 李名世

摘 要 在 VB环境下通过窗口子分类技术实现了一个能够获取所有 Windows消息的消息控件。文中提供了该控件的源代码,还介绍了 VB的一个未公开的函数。

主题词 窗口子分类 ActiveX VB

VB的编程途径,称为“事件驱动”编程,由事件驱动方式所开发的应用程序“对应”于在 Windows环境中发生的事件。但 VB也因此屏蔽了 Windows内含的消息驱动机制,在编程中只能对 VB提供的有限事件(对应于部分 Windows消息)进行编程,程序不能自由定义用户事件(对应特殊用户消息),这不能不说是 VB的一个缺点。为此作者利用 VB6.0设计了一个消息控件,它能够获取所有的 Windows消息,从而使 VB具备了 VC++、Delphi等语言才拥有的消息处理特征。

消息控件的设计原理是:采用窗口子分类技术,即用消息控件的窗口回调函数取代 VB缺省的窗口回调函数。当应用程序存在消息输入队列时,Windows将调用消息控件的窗口回调函数,在该窗口回调函数中,就可以截获所有的 Windows消息,并把该消息转化成消息事件,留给使用该控件的程序员处理。

在 VB控件设计环境下,窗口回调函数必须放在一个模块(module)内。要实现窗口子分类,还需要使用几个 WIN32 API函数,如下:

以下是位于 module内的代码。

```
Option Explicit
Const GWL_WNDPROC= (-4&)
                    '窗口回调函数地址所在处。
Declare Function SetWindowLong& Lib "user32" Alias "SetWindowLongA"_
    (By Val hWnd As Long By Val nIndex As Long By Val dwNewLong As Long)
'用 SetWindowLong API设置窗口的回调函数。
Declare Function CallWindowProc& Lib "user32" Alias "CallWindowProcA"_
    (By Val lpPrevWndFunc As Long By Val hWnd As Long)
    By Val MSG As Long By Val wParam As Long By Val lParam As Long)
'用 CallWindowProc API调用窗口可能存在的其它回调函数,本程序用该 API
'调用 VB缺省的窗口回调函数
Public Declare Sub CopyMemory Lib "Kernel32" Alias "RtlMoveMemory"_
    (pDest As Any, pSource As Any, ByVal ByteLen As Long)
'在 module中还申明了下述结构用于存储控件的使用情况、控件实例的地址、VB窗口句柄
'和 VB窗口缺省回调函数的地址
Type Instances
    in_use As Boolean       '控件是否正在使用
    ClassAddr As Long       '控件的地址
    hWnd As Long            'VB窗口句柄
    PrevWndProc As Long     'VB窗口缺省回调函数的地址
End Type
Public mInstance As instances
                            '创建一个 Instances的全局实例
Public IsHooked As Boolean
                            '存储窗口是否已经子分类
'以下为窗口回调函数 SwitchBoard的源代码,消息控件将用 SwitchBoard函数取代 VB
'缺省的窗口回调函数
'在 Switchboard函数中首先调用控件的消息处理函数,最后调用 VB缺省的窗口回调函数。
Public Function SwitchBoard(By Val hWnd As Long, ByVal MSG As Long, _
    By Val wParam As Long, By Val lParam As Long) As
```

[20] 杨 强 厦门大学计算机系 厦门 361005

```
Long
    Dim MyUCAsMsgHookCtrl"定交一个控件实例
    Dim Cancel As Boolean
    Dim PrevWndProc As Long
If IsHooked= True Then
    PrevWndProc= mInstance.PrevWndProc
                '获取 VB缺省窗口回调函数地址
End If
    On Error Resume Next
    CopyMemory MyUC, mInstance.ClassAddr, 4
                '调用 CopyMemory API把控件实例地址
    '赋值给 MyUC
    Cancel= False
    MyUC.HandleMsg hWnd, MSG, wParam, lParam
                '调用控件的 HandleMsg友元函数
    CopyMemory MyUC, 0&, 4    '清除 MyUC实例
If Cancel= False Then
     SwitchBoard= CallWindowProc(PrevWndProc, hWnd,
MSG, wParam, lParam)
     'SwitchBoard最后调用 VB缺省的窗口回源函数
End If
    End Function
'
-------------------------------------------------------
'Hook_Window 全局函数调用 SetWindowLong API实现窗口子分类
Public Sub Hook-Window(By Val hWnd As Long)
    If mInstance.in_use= True Then
      If mInstance.PrevWndProce= 0& Then
          mInstance.PrevWndProc = SetWindowLong
(hWnd, -
          GWL_WNDPROC, AddressOf Switch Board)'调
用 AddressOf函数获得
    'SwitchBoard函数的地址
    End If
    mInstance.hWnd= hWnd
    IsHooked= True
Else
    'Raise Error
End If
    End Sub
    '
-------------------------------------------------------
    'UnHookWindow 全局函数恢复窗口缺省的回调函数
    Public Sub UnHookWindow()
    If mInstance.in_use= True Then
      If mInstance.PrevWndProc<>0& Then
          SetWindowLong mInstance.hWnd, GWL_
WNDPROC, -
          mInstance.PrevWndProc
    mInstance.PrevWndProc= 0&
    mInstance.hWnd= 0&
    IsHooked= False
End If
    Else
    'Raise Error
End If
End Sub
```

本文所设计的控件命名为 MsgHookCtrl. 为该控件设计了两个方法(Method)和一个事件(Event): SetHook方法调用前文介绍的 Hook_Window 全局函数实现窗口的分子分类: SetUnhook方法调用 SetUnhook全局函数恢复 VB缺省的窗口回调函数; MessageProc事件在 HandleMsg函数中被触发, 它含有标准的 Windows消息参数, 从而使使用该消息控件的程序员能够获取所有的 Windows消息.

在控件的 Initialize事件中调用了 VB的一个未分开的函数 ObjPtr. 该函数可以获取一个对象实例的 32位线性地址.

```
'
-------------------------------------------------------
Public Event MessageProc(By Val hWnd As Long, By Val
MSG As Long, -
    By Val wParam As Long, By Val lParam As Long)'申明
MessageProc事件
Private Sub UserControl_Initialize()
If mInstance.in_use= False Then
     "MsgBox "OK"
    mInstance.in_use= True
    '保证了在一个程序中只能有一个消息控件发挥作用
    mInstance.ClassAddr= ObjPtr(Me)
    '在 mInstance中存储控件实例的地址
Else
    'Raise Error
End If
'
-------------------------------------------------------
End Sub
Private Sub UserControl_Terminate()
    If IsHooked= True Then
    UnHookWindow
    End If
End Sub
```

(下转第 47页)

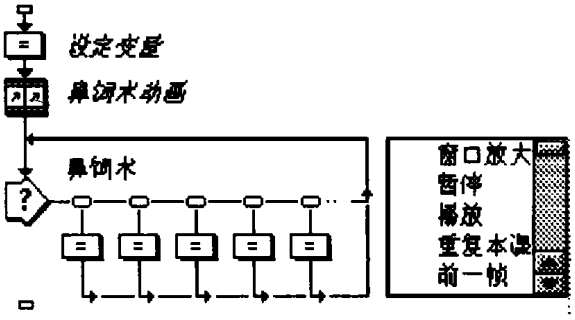

图 1

Scale(2, W indow H andle)
计算图标“暂停”的内容设置
pause = False
M ediaPause(IconID@“ flcl”, Pause)
Pause(W indowH andle)
计算图标“播放”的内容设置:
Resum e(w indowH andle)
Pause = FA LSE
M ediaPause(IconID@“ flcl”, Pause)
计算图标“重复本课内容”的内容设置:
C lose( )
Go to(IconID@“鼻饲术动画”).

四. 结论

本课件脚本理论性、系统性强. 它以优秀的软件为开发平台, 操作明了、简便. 三维动画以生动、形象、逼真的效果, 引起学生的极大兴趣. 本课件可广泛应用于护校《基础护理学》的计算机辅助教学, 也可广泛应用于医院的在职护士的护理操作培训. 如今计算机已广泛应用于各个行业, 我们医务工作者应利用现有的高新技术为古老医学开拓教学、科研的领域.

参考文献

[1]甘登岱等编, Auto CAD 12 0使用大全, 学苑出版社, 1993, 12
[2]中林等编, 3D S tudio R4 0实用教程, 学苑出版社, 1994, 8
[3]廖凯等编, A dobe P rem iere R4. 0软件操作手册, 上海科技出版社, 1996, 11
[4]达米教室, A uthorw are 3. 0使用指南 (上、下), 清华大学出版社
[5]郭启全、李燕, 多媒体 CA I的创作方法和实例, 电子工业出版社
[6]林启垄, V isual BASIC 4 0, 科学出版社

(收稿日期: 99年 2月 4日)

(上接第 52页)

```
Public Sub SetHook(By Val hWnd As Long)

Hook-Window hWnd
End Sub
Public Sub SetUnhook( )
    UnHookWindow
End Sub
‘ HandleMsg是一个有元函数, 该函数在 SwitchBoard中被控件实例调用.
‘ 在 HandleMsg中用 RaiseEvent触发 MseeageProc事件, 使程序员能够对消息进行编程
Friend Sub HandleMsg(By Val hWnd As Long, By Val MSG As Long, _
    By Valw Param As Long By Val lParam As Long)
    RaiseEvent MessageProc(hWnd, MSG, wParam, lParam)
End Sub
```

因为 VB已经把大部分的消息实现为预定义的事件, 所以可在 SwitchBoard回调函数中进一步过滤出感兴趣的消息, 即把那些能够形成 VB事件的消息交由 VB的缺省回调函数处理, 而把 VB没有形成事件而程序员又需要处理的消息通过控件的 MessageProc通知给程序员.

参考文献

[1]Brian Siler Jeff Spotts著, 康博创作室译, 《V isual Basic 6 开发使用指南》, 机械工业出版社, 1999年 4月.
[2]T im K ilgore著: “The Sw itchboard: A m ethod for handling subclassing in ActiveX controls”, W eb Pages on the Internet, 1997年 5月.

(收稿日期: 99年 4月 5日)